# RPC BROKER

# SYSTEMS MANUAL

Version 1.1; Patch XWB*1.1*40

September 1997

Revised February 2005

Department of Veterans Affairs
VistA Health Systems Design & Development (HSD&D)
Infrastructure and Security Services (ISS)

# Revision History

## Documentation Revisions

The following table displays the revision history for this document. Revisions to the documentation are based on patches and new versions released to the field.

| Date | Revision | Description | Author(s) |
|---|---|---|---|
| 09/97 | 1.0 | Initial RPC Broker Version 1.1 software release. | Thom Blom and Joel Ivey, Oakland OIFO |
| 04/08/02 | 2.0 | Revised Version for RPC Broker Patch XWB*1.1*13. | Thom Blom and Joel Ivey, Oakland OIFO |
| 05/08/02 | 3.0 | Revised Version for RPC Broker Patch XWB*1.1*26. | Thom Blom and Joel Ivey, Oakland OIFO |
| 02/23/05 | 4.0 | Revised Version for RPC Broker Patches XWB*1.1*35 and 40.<br>Also, reviewed document and edited for the "Data Scrubbing" and the "PDF 508 Compliance" projects.<br>**Data Scrubbing—**Changed all patient/user TEST data to conform to HSD&D standards and conventions as indicated below:<br>• The first three digits (prefix) of any Social Security Numbers (SSN) start with "000" or "666."<br>• Patient or user names are formatted as follows: NHEPATIENT,[N] or NHEUSER,[N] respectively, where the N is a number written out and incremented with each new entry (e.g., NHEPATIENT, ONE, NHEPATIENT, TWO, etc.).<br>• Other personal demographic-related data (e.g., addresses, phones, IP addresses, etc.) were also changed to be generic.<br>**PDF 508 Compliance—**The final PDF document was recreated and now supports the minimum requirements to be 508 compliant (i.e., accessibility tags, language selection, alternate text for all images/icons, fully functional Web links, successfully passed Adobe Acrobat | Thom Blom and Joel Ivey, Oakland OIFO |

| Date | Revision | Description | Author(s) |
|---|---|---|---|
| | | Quick Check). | |

**Table i: Documentation revision history**

# Patch Revisions

For the current patch history related to this software, please refer to the Patch Module on FORUM.

# Contents

# Figures and Table

# Acknowledgements

The RPC Broker Development Team consists of the following Development and Infrastructure Service (DaIS) and Infrastructure & Security Services (ISS) personnel (listed alphabetically):

- ISS Program and Project Manager—Larry Weldon
- Centralized Planner Support Team (CPST)—Laura Rowland
- Developers—Alan Chan, Wally Fort, Joel Ivey (lead), and Raul Mendoza
- Functional Analysts—Lauren Hardeen and Lauren Gorgoglione
- Software Quality Assurance (SQA)—Matt Alderman
- Technical Writer—Thom Blom

The RPC Broker Development Team would like to thank the following sites/organizations/personnel for their assistance in reviewing and/or testing RPC Broker V. 1.1, Patch XWB*1.1*40 software and documentation (listed alphabetically):

- Computerized Patient Record System (CPRS) GUI—Development Team

# Orientation

## How to Use this Manual

Throughout this manual, advice and instructions are offered regarding the use of the RPC Broker V. 1.1 and the functionality it provides for Veterans Health Information Systems and Technology Architecture (VistA).

There are no special legal requirements involved in the use of the RPC Broker.

This manual uses several methods to highlight different aspects of the material:

- Various symbols are used throughout the documentation to alert the reader to special information. The following table gives a description of each of these symbols:

| Symbol | Description |
|---|---|
| ℹ | Used to inform the reader of general information including references to additional reading material |
| ⚠ | Used to caution the reader to take special notice of critical information |

**Table ii: Documentation symbol descriptions**

- Descriptive text is presented in a proportional font (as represented by this font).
- Conventions for displaying TEST data in this document are as follows:
    - ➢ The first three digits (prefix) of any Social Security Numbers (SSN) will begin with either "000" or "666."
    - ➢ Patient and user names will be formatted as follows: [Application Name]PATIENT,[N] and [Application Name]USER,[N] respectively, where "Application Name" is defined in the Approved Application Abbreviations document and "N" represents the first name as a number spelled out and incremented with each new entry. For example, in Kernel (KRN) test patient and user names would be documented as follows: KRNPATIENT,ONE; KRNPATIENT,TWO; KRNPATIENT,THREE; etc.
- Sample HL7 messages, "snapshots" of computer online displays (i.e., roll-and-scroll screen captures/dialogues) and computer source code, if any, are shown in a *non*-proportional font and enclosed within a box.

  Also included are Graphical User Interface (GUI) Microsoft Windows images (i.e., dialogues or forms).

    - ➢ User's responses to online prompts will be boldface.
    - ➢ The "**<Enter>**" found within these snapshots indicate that the user should press the Enter key on their keyboard. Other special keys are represented within < > angle brackets. For example, pressing the PF1 key can be represented as pressing **<PF1>**.

- ➢ Author's comments, if any, are displayed in italics or as "callout" boxes.

  Callout boxes refer to labels or descriptions usually enclosed within a box, which point to specific areas of a displayed image.

- All uppercase is reserved for the representation of M code, variable names, or the formal name of options, field and file names, and security keys (e.g., the XUPROGMODE key).
- Object Pascal code uses a combination of upper- and lowercase characters. All Object Pascal reserved words are in boldface type.

## Commonly Used Terms

The following is a list of terms and their descriptions that you may find helpful while reading the RPC Broker documentation:

| Term | Description |
|---|---|
| **Client** | A single term used interchangeably to refer to a user, the workstation (i.e., PC), and the portion of the program that runs on the workstation. |
| **Component** | A software object that contains data and code. A component may or may not be visible.<br>For a more detailed description, please refer to the *Borland Delphi for Windows User Guide*. |
| **GUI** | The Graphical User Interface application that is developed for the client workstation. |
| **Host** | The term Host is used interchangeably with the term Server. |
| **Server** | The computer where the data and the RPC Broker remote procedure calls (RPCs) reside. |

**Table iii: Commonly used RPC Broker terms**

Please refer to the "Glossary" for additional terms and definitions.

## How to Obtain Technical Information Online

Exported file, routine, and global documentation can be generated through the use of Kernel, MailMan, and VA FileMan utilities.

Methods of obtaining specific technical information online will be indicated where applicable under the appropriate topic.

Please refer to the *RPC Broker Technical Manual* for further information.

### Help at Prompts

VistA software provides online help and commonly used system default prompts. Users are encouraged to enter question marks at any response prompt. At the end of the help display, you are immediately returned to the point from which you started. This is an easy way to learn about any aspect of VistA software.

To retrieve online documentation in the form of Help in any VistA character-based product:

- Enter a single question mark ("**?**") at a field/prompt to obtain a brief description. If a field is a pointer, entering one question mark ("**?**") displays the HELP PROMPT field contents and a list of choices, if the list is short. If the list is long, the user will be asked if the entire list should be displayed. A **YES** response will invoke the display. The display can be given a starting point by prefacing the starting point with an up-arrow ("^") as a response. For example, **^M** would start an alphabetic listing at the letter M instead of the letter A while **^127** would start any listing at the 127th entry.
- Enter two question marks ("**??**") at a field/prompt for a more detailed description. Also, if a field is a pointer, entering two question marks displays the HELP PROMPT field contents and the list of choices.
- Enter three question marks ("**???**") at a field/prompt to invoke any additional Help text stored in Help Frames.

### Obtaining Data Dictionary Listings

Technical information about files and the fields in files is stored in data dictionaries. You can use the List File Attributes option on the Data Dictionary Utilities submenu in VA FileMan to print formatted data dictionaries.

For details about obtaining data dictionaries and about the formats available, please refer to the "List File Attributes" chapter in the "File Management" section of the *VA FileMan Advanced User Manual*.

## Assumptions About the Reader

This manual is written with the assumption that the reader is familiar with the following:

- VistA computing environment:
  - Kernel
  - VA FileMan data structures and terminology
- Microsoft Windows environment
- M programming language
- Object Pascal programming language.
- Borland Delphi Integrated Development Environment (IDE)

Itl provides an overall explanation of configuring RPC Broker and the functionality contained in RPC Broker Version 1.1. However, no attempt is made to explain how the overall VistA programming system is integrated and maintained. Such methods and procedures are documented elsewhere. We suggest you look at the various VA home pages on the World Wide Web for a general orientation to VistA. For example, go to the Health Systems Design & Development (HSD&D) Home Page at the following Web address:

http://vista.med.va.gov/

## Reference Materials

Readers who wish to learn more about the RPC Broker should consult the following:

- *RPC Broker Release Notes*
- *RPC Broker Installation Guide*
- *RPC Broker Systems Manual* (this manual)
- *RPC Broker Technical Manual*
- *RPC Broker Getting Started with the Broker Development Kit (BDK)*
- *RPC Broker Developer's Guide* (i.e., BROKER.HLP, online help designed for programmers, distributed in the BDK)
- RPC Broker Home Page at the following Web address:

  http://vista.med.va.gov/broker/index.asp

  This site provides announcements, additional information (e.g., Frequently Asked Questions [FAQs], advisories), documentation links, archives of older documentation and software downloads.

VistA documentation is made available online in Microsoft Word format and in Adobe Acrobat Portable Document Format (PDF). The PDF documents *must* be read using the Adobe Acrobat Reader (i.e., ACROREAD.EXE), which is freely distributed by Adobe Systems Incorporated at the following web address:

http://www.adobe.com/

For more information on the use of the Adobe Acrobat Reader, please refer to the *Adobe Acrobat Quick Guide* at the following web address:

http://vista.med.va.gov/iss/acrobat/index.asp

VistA documentation can be downloaded from the Health Systems Design and Development (HSD&D) VistA Documentation Library (VDL) Web site:

http://www.va.gov/vdl/

VistA documentation and software can also be downloaded from the Enterprise VistA Support (EVS) anonymous directories:

| | |
|---|---|
| • Albany OIFO | ftp.fo-albany.med.va.gov |
| • Hines OIFO | ftp.fo-hines.med.va.gov |
| • Salt Lake City OIFO | ftp.fo-slc.med.va.gov |
| • Preferred Method | download.vista.med.va.gov |

This method transmits the files from the first available FTP server.

**DISCLAIMER: The appearance of external hyperlink references in this manual does *not* constitute endorsement by the Department of Veterans Affairs (VA) of this Web site or the information, products, or services contained therein. The VA does *not* exercise any editorial control over the information you may find at these locations. Such links are provided and are consistent with the stated purpose of this VA Intranet Service.**

# 1. Introduction

## Overview

The Remote Procedure Call (RPC) Broker (also referred to as "Broker") is a client/server system within VA's Veterans Health Information Systems and Technology Architecture (VistA) environment. It establishes a common and consistent foundation for client/server applications being written as part of VistA. It enables client applications to communicate and exchange data with M Servers.

The RPC Broker is a bridge connecting the client application front-end on the workstation (e.g., Delphi GUI applications) to the VistA M-based data and business rules on the server. It links one part of a program running on a workstation to its counterpart on the server. Therefore, the RPC Broker assists in opening the traditionally proprietary VistA software to Commercial Off-the-Shelf (COTS) and Hybrid Open Systems Technology (HOST) products.

This manual provides descriptive information and instructions on the use of the RPC Broker client/server software. The emphasis is on the use of Borland's Delphi software. However, the RPC Broker does support other client environments.

This document is intended for the VistA development community, the Information Resource Management (IRM) staff, and clinicians using Broker-based client/server applications. A wider audience of technical personnel engaged in operating and maintaining the Department of Veterans Affairs (VA) software may also find it useful as a reference.

**The RPC Broker includes the following:**

- A common communications driver interface that handles the device-specific characteristics of the supported communications protocol.
- An interface component separate from the communications driver that interprets the message, executes the required code, and eventually returns data to the communications driver.
- A common file that all applications use to store the information on the queries to which they respond (i.e., REMOTE PROCEDURE file [#8994]).
- Architecture that supports multiple GUI and client front-ends.

This version of the Broker also includes the Broker Development Kit (BDK). The BDK provides VistA application programmers with the following features:

- The capability to create GUI client/server VistA applications using Borland's Delphi software. The BDK provides the TRPCBroker, TSharedRPCBroker, and TXWBRichEdit components, which developers use in Delphi applications to execute remote procedure calls (RPCs) on VistA M Servers.
- Support for COTS/HOST client/server software using the Broker Dynamic Link Library (DLL).

**The RPC Broker:**

- Operates in a 32-bit environment while supporting VistA applications previously developed in the 16-bit environment (e.g., PCMM). The client workstation can be running any of the following Microsoft operating systems:
  - Windows 2000
  - Windows XP
- Provides support for Single Sign-On/User Context. As of Patch XWB*1.1*40, the TCCOWRPCBroker component enabled Single Sign-On/User Context (SSO/UC) in CCOW-enabled applications. This allow users to authenticate and sign on to multiple applications that are CCOW-enabled and SSO/UC-aware using a single set of credentials, which reduces the need for multiple ID's and passwords in the Health*e*Vet clinician desktop environment.

  For more information on SSO/UC, please consult the *Single Sign-On/User Context (SSO/UC) Installation Guide* and *Single Sig*

- Provides support for Non-Callback Connections. As of Patch XWB*1.1*35, the RPC Broker components are built with a UCX or non-callback Broker connection, so that it can be used from behind firewalls, routers, etc. This functionality is controlled via the new TRPCBroker component IsBackwardCompatibleConnection property.
- Provides support for Silent Logons. As of Patch XWB*1.1*13, the RPC Broker provides "Silent Login" capability. It provides functionality associated with the ability to make logins to a VistA M Server without the RPC Broker asking for Access and Verify code information.
- Provides support for Auto Signon. Users need only sign on once when accessing both a VistA roll-and-scroll (e.g., Laboratory, Pharmacy) and a VistA client/server GUI-based application (e.g., CPRS, NOIS, PCMM) on the same workstation, regardless of which application is started first.

  For more information on Auto Signon, please refer to the "Integrated Auto Signon for Multiple User Sessions" topic in Chapter 2, "System Features" in this manual.

- Allows multi-instances of the RPC Broker. As of Patch XWB*1.1*13, the RPC Broker code was modified to permit an application to open two separate Broker instances with the same Server/ListenerPort combination, resulting in two separate partitions on the server. Previously, an attempt to open a second Broker instance ended up using the same partition. For this capability to be useful for concurrent processing, an application would have to use threads to handle the separate Broker sessions

  **Although we believe there should be no problems, the RPC Broker is not yet guaranteed to be thread safe.**

- Provides enhanced Broker management and configuration tools (e.g., debugging tools, RPC BROKER SITE PARAMETERS file (#8994.1), enhanced Broker Listener).

  For more information on troubleshooting the Broker, please refer to Chapter 4, "Troubleshooting" in this manual.

## How Does It All Work?

The process begins on a user's workstation (i.e., PC), running Microsoft Windows, which is either connected directly or remotely via a modem to a site's local area network (LAN). The workstation must be able to run some version of Transmission Control Protocol/Internet Protocol (TCP/IP).

For more specific environment requirements, please refer to the *RPC Broker Installation Guide*.

Currently only Winsock-compliant TCP/IP protocol is supported on the LAN or remotely as Point-to-Point Protocol (PPP) or Serial Line Internet Protocol (SLIP). You must use RAS (Remote Access Service) or Dialup Networking to connect to the server using PPP or SLIP. For the setup of RAS or Dialup Networking, please refer to the appropriate operating system's documentation.

When a user starts a VistA program on the client, the program requests a connection with a server. The server is continuously running at least one Broker "Listener" job in the background whose sole purpose is to establish connections with clients.

Once the Listener receives a connection request, it does the following:

1. Validates the message.
2. Creates (spawns, jobs off) another process "Handler." The Handler process does the work to satisfy the client's requests.
3. Goes back to listening.

When the connection to the server is established, users who are not already logged into the server are asked to identify themselves by logging in with their Access and Verify codes. With the implementation of Auto Signon, users are considered already logged in to the server if they have previously logged in to a VistA GUI or roll-and-scroll application that is still running on their workstation. After a successful login, the application is active on both the server and the client.

For more information on Auto Signon, please refer to the "Integrated Auto Signon for Multiple User Sessions" topic in Chapter 2, "System Features" in this manual.

As you manipulate the interface, your client process is reading and writing data to the server. The reading and writing is carried out as messages traveling over the TCP/IP link. In the message sent to the server, client applications will include the name of the requested RPC to be activated and its associated parameters. These RPCs will be written in M and registered in a file containing available and authorized RPCs (i.e., REMOTE PROCEDURE file [#8994]). Upon receipt by the server, the message is decoded, the requested remote procedure call is activated, and the results are returned to the calling application.

The server receives a message from the client and parses out the name of the remote procedure call and its parameters. The Broker module on the server looks up the remote procedure call in the REMOTE PROCEDURE file (#8994), verifies that the RPC is allowed to run in the context of the application, and executes the RPC using the passed parameters. At this point, the server side of the application processes the request and returns the result of the operation. The result of the call contains either several values or a single value. If the operation is a query, then the result is a set of records that satisfy that query. If the operation is to simply file the data on the server or it is unnecessary to return any information, then, typically, notification of the success of the operation will be returned to the client.

This version of the RPC Broker supports messaging for non-Delphi client applications (e.g., Borland C++, Microsoft Visual Basic, or other COTS Microsoft Windows-based products). The RPC Broker Version 1.1 supplies a set of functions providing a Dynamic Link Library (DLL) interface that allows non-Delphi applications to conform to the client side interface of the Broker.

For more specific information about the Broker DLLs, please refer to the *RPC Broker Developer's Guide* (i.e., BROKER.HLP, online help in the BDK).

## System Overview

The following diagram gives an overview of the VistA/RPC Broker environment:

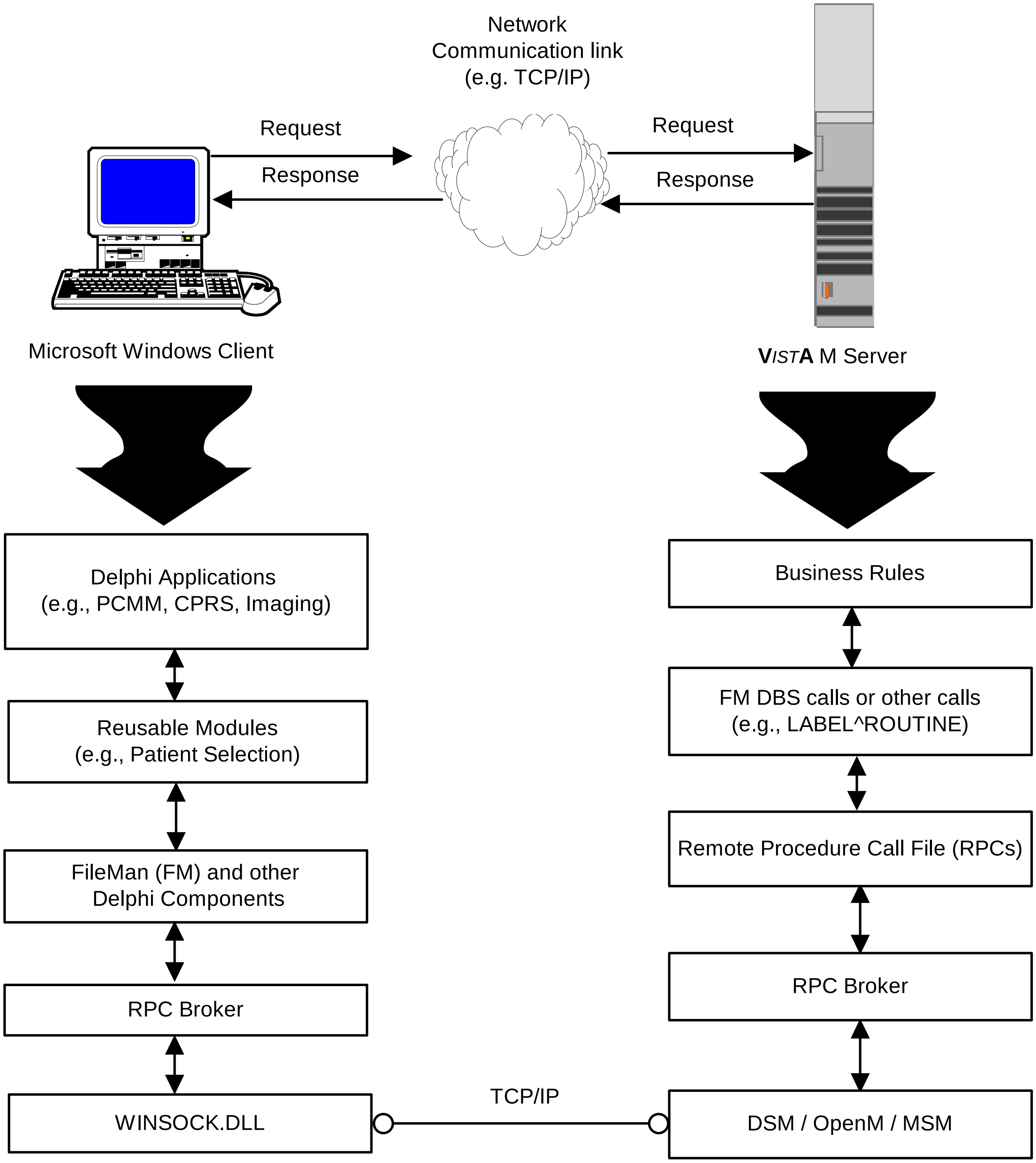


**Figure 1-1: VistA RPC Broker system overview diagram**

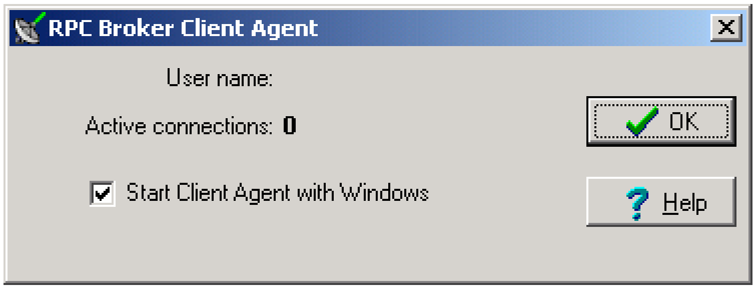
# 2. System Features

## Client Features

### RPC Broker Client Agent

The RPC Broker Client Agent program (i.e., CLAGENT.EXE) runs in support of the Auto Signon process. This program automatically and continuously runs in the background on the client workstation and normally should *not* be closed or shut down by the user. A satellite dish icon will be displayed in the System Tray indicating the Broker Client Agent is running. The icon will change when an active connection is made to the server—a green line indicating an active connection will emanate from the satellite dish.

By double clicking on the Client Agent icon, you can see how many active connections are currently open, as shown below. However, the "Active connections" count may include "orphan" connections that are no longer active. Use this count as an approximate count only.



User name:

Active connections: 0

☑ Start Client Agent with Windows

**Figure 2-1: RPC Broker Client Agent dialogue**

The "Start Client Agent with Windows" checkbox should be checked so that Auto Signon, if allowed, will be operational. By default, this box is checked. However, if a particular workstation is not always connected to the network upon startup, you may wish to prevent the Client Agent from starting automatically. You can always reset it to start automatically by starting the Client Agent manually first and re-checking this checkbox.

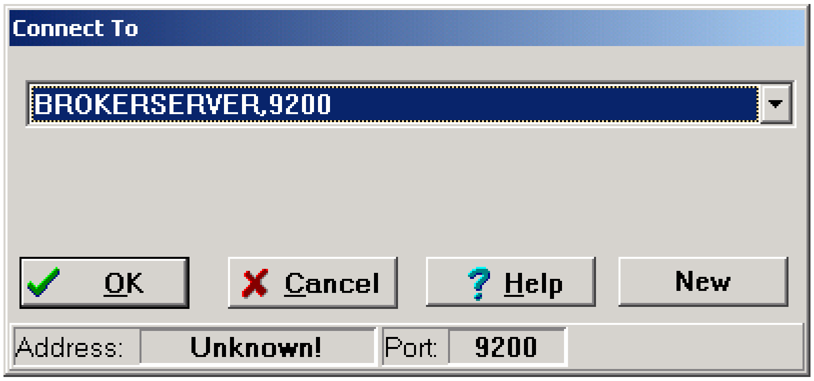
For more information on Auto Signon, please refer to the "Integrated Auto Signon for Multiple User Sessions" topic that follows in this chapter.

The RPC Broker Client Agent is installed with the End-User Client Workstation installation of the RPC Broker and is *not* included with RPC Broker Development Kit (BDK).

For more information on the End-User Client Workstation and Client Agent installation, please refer to the *RPC Broker Installation Guide*.

## "Connect To" Dialogue

Upon logging in to a VistA client/server application, users may be presented with the "Connect To" dialogue, as shown below:



BROKERSERVER,9200

Address: Unknown! Port: 9200

**Figure 2-2: Server and port configuration selection dialogue**

This server and port configuration selection dialogue can be used by Delphi VistA client/server applications that wish to allow users to:

- Select an existing server name and associated port from a list of servers entered into the Microsoft Windows Registry
- Enter a new server name, Internet Protocol (IP) address, and associated port number.

For example, this can be useful when you want to run the application in either a Test or Production account.

To add a new server and associated port number to the Microsoft Windows Registry, press the New button (see Figure 2-2 above). You are presented with the "Add Server" dialogue, as shown below:

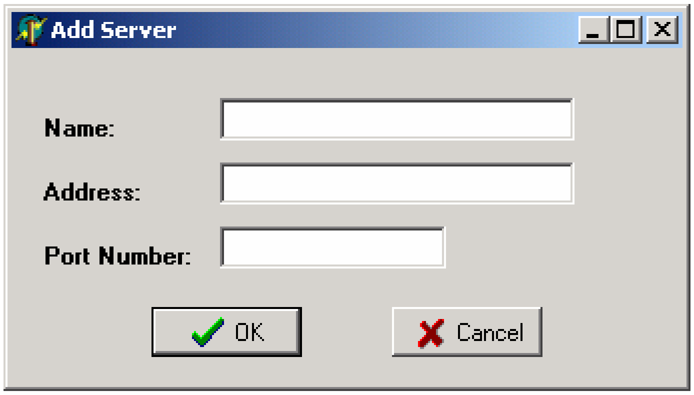


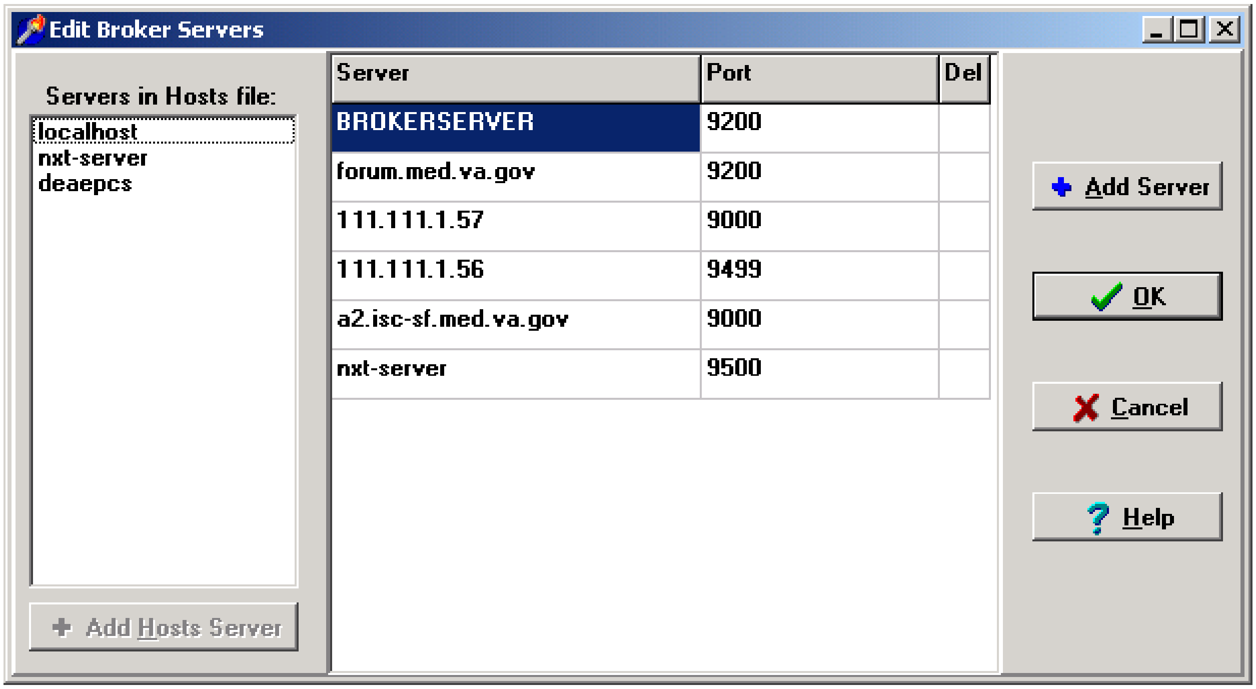
**Figure 2-3: Add Server dialogue**

You can also add additional server names and ports to the Microsoft Windows Registry by using the Edit Broker Servers program.

For more information on adding new servers, please refer to the "Edit Broker Servers Program" topic that follows.

## Edit Broker Servers Program

If someone in IRM wishes to add, modify, or delete servers and ports to be used by the Broker, they can run the Edit Broker Servers program (i.e., SERVERLIST.EXE), as shown below:



Servers in Hosts file:

localhost
nxt-server
deaepcs

| Server | Port | Del |
|---|---|---|
| BROKERSERVER | 9200 | |
| forum.med.va.gov | 9200 | |
| 111.111.1.57 | 9000 | |
| 111.111.1.56 | 9499 | |
| a2.isc-sf.med.va.gov | 9000 | |
| nxt-server | 9500 | |

**Figure 2-4: Edit Broker Servers dialogue**

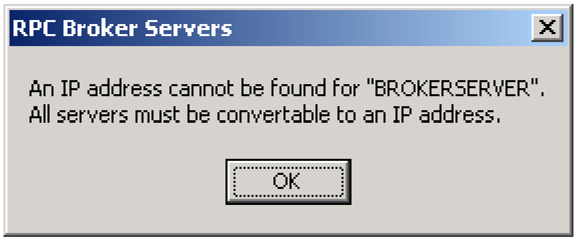
Use this program to modify or add Listeners/Ports to the Microsoft Windows Registry. SERVERLIST.EXE can be copied to any workstation for this purpose.

This program only displays HOSTS file entries; it does not edit the HOSTS file.

### Adding Entries

You are given two methods of adding new server entries to the Microsoft Windows Registry using the Edit Broker Servers program:

1. **Choose From an Existing List—**A list of servers available from your HOSTS file is displayed in the "Servers in Hosts file" list box on the left of the dialogue (see Figure 2-4). Thus, you don't have to remember and type the server names yourself. Select one or more from the list and press the "+Add Hosts Server" button in the lower left of the dialogue. This creates the new grid line(s) with the server(s) selected automatically "stuffed" into the Server cell(s). Complete each new entry by typing in the appropriate port(s). When finished, press OK.
2. **Add a New Server—**Alternatively, you can press the "+Add Server" button on the right of the dialogue (see Figure 2-4). This creates a new grid line. Enter the name of the server you want added. Complete each new entry by typing in the appropriate port(s). When finished, press OK. The program will attempt to resolve the server name to an IP address either through the Domain Name Service (DNS) or by looking it up in the HOSTS file on the client workstation. If this is successful, the new entry will be added to the Microsoft Windows Registry. If the server name cannot be resolved, an error message will be displayed and you will have to correct your entry, as shown below:



An IP address cannot be found for "BROKERSERVER".
All servers must be convertable to an IP address.

**Figure 2-5: Sample error message when adding a new server entry**

Hint: If you're running a PC Network with Microsoft Windows NT, the BROKERSERVER added to the Services list on the NT network will speed up client access times (i.e., keeps it from having to do a double lookup with first IP then service, it merely looks at the Services list).

### Modifying Entries

In order to modify or change a server or port, simply place the cursor in the appropriate Server or Port field and make the change (see Figure 2-4). When finished, press OK.

Server names must be resolvable through DNS or the HOSTS file.

### Deleting Entries

In order to delete a pre-existing entry, just click in the Del column. An asterisk appears in the Del column signifying a deletion. Another click toggles the deletion off. When finished, press OK.

## Standalone Programs and their Associated Help Files

Each of the following standalone Broker programs, distributed with this version of the Broker, have an associated help file that must reside in the *same* directory in order to provide online help for that particular standalone program:

| Standalone Program | Associated Help File | Location |
|---|---|---|
| BROKERPROGPREF.EXE | BROKERPROGPREF.HLP | Programmer Workstation (BDK) |
| CLAGENT.EXE | CLAGENT.HLP | End-User Workstation |
| RPCTEST.EXE | RPCTEST.HLP | End-User Workstation |
| SERVERLIST.EXE | SERVERLIST.HLP | Programmer Workstation (BDK) |

**Table 2-1: Standalone RPC Broker programs and their associated help files**

The installation of the Broker will automatically load these associated files into the appropriate directories. If you choose to "export" a standalone program (e.g., SERVERLIST.EXE) to another client workstation, make sure you include its associated help file and place them both in the *same* directory.

For more information on the BROKERPROGPREF.EXE, please refer to the *RPC Broker Developer's Guide* (i.e., BROKER.HLP, online help designed for programmers, distributed with the BDK).

For more information on the CLAGENT.EXE, please refer to the "RPC Broker Client Agent" topic previously described in this chapter.

For more information on the RPCTEST.EXE, please refer to Chapter 4, "Troubleshooting" in this manual.

For more information on the SERVERLIST.EXE, please refer to the "Edit Broker Servers Program" topic previously described in this chapter.

## HOSTS File

The HOSTS file is an ASCII text file that contains a list of the servers and their IP addresses. However, use of the HOSTS file is *not* a requirement for the Broker. The use of the HOSTS file depends on the way the local area network (LAN) is implemented and managed at a site. Clients can bypass the HOSTS file and use DNS, DHCP (Dynamic Host Configuration Protocol), or WINS (Windows Name Service).

To modify or add servers to the HOSTS file, edit the file using a text editor (e.g., Microsoft Notepad).

The following table illustrates where you can find this file based on your client Microsoft Windows operating system (OS):

| Version of Windows OS | File (Location and Name) |
|---|---|
| Windows XP | C:\WINDOWS\SYSTEM32\DRIVERS\ETC\HOSTS |
| Windows 2000 | C:\WINDOWS\SYSTEM32\DRIVERS\ETC\HOSTS |
| Windows NT 4.0 | C:\WINNT\SYSTEM32\DRIVERS\ETC\HOSTS |
| Windows NT 3.51 | C:\WINDOWS\SYSTEM32\DRIVERS\ETC\HOSTS |
| Windows 98 | C:\WINDOWS\HOSTS |
| Windows 95 | C:\WINDOWS\HOSTS |

**Table 2-2: HOSTS file location in Windows**

A sample of the Microsoft Windows XP HOSTS file (i.e., C:\WINDOWS\SYSTEM32\DRIVERS\ETC\HOSTS) is displayed below (modifications/additions made to this sample file are in boldface and italicized):

```
# Copyright (c) 1993-1999 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97     rhino.acme.com          # source server
#       38.25.63.10     x.acme.com              # x client host
#
#   IP            Host
# ADDRESS         Name            Description
#    |              |                  |
#    |              |                  |
#    V              V                  V

127.0.0.1       localhost       # loopback
192.1.1.1       BROKERSERVER    # Broker Server
```

**Figure 2-6: Sample HOSTS file**

The last entry in this file (i.e., BROKERSERVER) was added to the sample HOSTS file for illustration purposes. We recommend you put in an entry that points to the main server you intend using with the Broker the majority of the time (e.g., BROKERSERVER). VistA applications can specify any server they wish.

**A DHCPSERVER entry is still required for software that uses Version 1.0 of the Broker. You may want to create an additional entry for BROKERSERVER in your HOSTS file or DNS. However, do not remove the DHCPSERVER entry already present.**

## Adding Entries

To add entries in the HOSTS file use a text editor (e.g., Microsoft's Notepad) to open the HOSTS file.

1. Move the cursor to the end of the last line displayed in the file.
2. Press the **<Enter>** key to create a new line.
3. On the new line, enter the desired IP address beginning in the first column, as described in the sample HOSTS file (see Figure 2-6). As recommended, add an appropriate IP address for the BROKERSERVER Host name as the next entry below "127.0.0.1".
4. After typing the IP address, type at least one space and enter the Host name that corresponds to that IP address. As recommended, type in BROKERSERVER as the next entry below "loopback".

   For example, the entry for a server at your site with an IP address of 192.1.1.1 would look like the following:

   ```
   127.0.0.1    localhost     # loopback      <---existing entry
   192.1.1.1    BROKERSERVER  # Broker Server <---added entry
   ```

5. Repeat Steps #1 - #4 until you have entered all of the IP addresses and corresponding Host names you wish to enter.
6. When your entries are complete, save the HOSTS file.

   Do not save the HOSTS file with an extension!

7. Close the HOSTS file and text editor.

## Modifying Entries

To modify entries in the HOSTS file use a text editor (e.g., Microsoft's Notepad) to open the HOSTS file.

1. Move the cursor to the line to be modified.
2. Modify the IP address, Host name, or both.
   - Make sure that at least one space separates the IP address from the corresponding Host name.
   - Make sure you have an entry for BROKERSERVER in this file.
3. Repeat Steps #1 - #2 until you have modified all of the IP addresses and corresponding Host names you wish to change.
4. When your entries are complete, save the HOSTS file.

   Do not save the HOSTS file with an extension!

5. Close the HOSTS file and the text editor.

## What Happened to the Client Manager?

The Client Manager, previously distributed with version 1.0 of the Broker, is no longer used by this version of the Broker. In version 1.0 of the Broker, the Client Manager provided two types of services:

1. It was used to invoke the RPCBI.DLL.
2. It was used by developers to set programmer preferences for using the original TRPCBroker component.

The RPCBI.DLL that was distributed with the RPC Broker V. 1.0 is no longer used, thus, the Client Manager is no longer required with this version of the Broker. Configuration of programmer preferences will now be done via the Broker Programmer Preferences dialogue.

For more information on the Broker Programmer Preferences dialogue, please refer to the "RPC Broker Programmer Preference Editor" topic in the *RPC Broker Developer's Guide* (i.e., BROKER.HLP, online help in the BDK).

**The RPCBI.DLL and Client Manager (i.e., CLMAN.EXE) installed with Broker V. 1.0 must not be removed from the VISTA/Broker directory on the client workstation. They are still required for 16-bit Broker-based applications created using version 1.0 of the Broker (e.g., PCMM).**

## What Happened to the VISTA.INI File?

The VISTA.INI file is no longer used by applications built with Version 1.1 of the Broker. However, this file will continue to be used by applications built using version 1.0 of the Broker (e.g., PCMM). During the installation of the Broker, relevant data from the VISTA.INI file will be moved to the Microsoft Windows Registry. Subsequent reads and writes will be done via the Registry.

**The VISTA.INI file created with Broker V. 1.0 must *not* be removed from the Microsoft Windows directory on the client workstation. It is still required for 16-bit Broker-based applications created using version 1.0 of the Broker (e.g., PCMM).**

The following are a list of items from the VISTA.INI file and their disposition with Version 1.1 of the Broker:

| VISTA.INI File Item | Disposition |
|---|---|
| ClientManagerPath<br>ErrorRetry<br>ClientManagerState | Client Manager items—*Not* moved to the Microsoft Windows Registry. |
| IdeConnect<br>ClearParameters<br>ClearResults<br>ListenerPort<br>Server | Programmer items—Moved to the Microsoft Windows Registry via a developer workstation installation (to be edited by the new Configuration form). |
| SignonPos<br>SignonSiz<br>IntroBackCol<br>IntroTextFont | Sign-on items—Moved to the Microsoft Windows Registry (these did *not* exist in version 1.0). These items will now be edited from the Sign-on form.<br>For more information, please refer to the "Users Can Customize VistA Signon Dialogue" topic in Chapter 3 in this manual. |
| HostsPath | No longer useful (i.e., Broker V. 1.1 Delphi code will *not* reference it). |
| [RPCBroker_Servers] section | Server/Port pairs—moved to the Microsoft Windows Registry via general workstation installations. These entries will now be edited via the Edit Broker Servers application.<br>For more information, please refer to the "Edit Broker Servers Program" topic in this chapter. |

**Table 2-3: VISTA.INI entries and Microsoft Windows Registry disposition table**

# Server Features

## Menu for System Managers

Patch XWB*1.1*9 introduced a menu for system managers [XWB MENU]:

```
Select RPC Broker Management Menu Option:

          RPC Listener Edit
          Start All RPC Broker Listeners
          Stop All RPC Broker listeners
```

**Figure 2-7: RPC Broker Management Menu option [XWB MENU]**

## Broker Listeners and Ports

You can run:

- A *single* Broker Listener, running on any available port.
- *Multiple* Broker Listeners running on the same IP address/CPU, but listening on *different* ports.
- *Multiple* Broker Listeners in the *same* UCI-volume, but on *different* IP addresses/CPUs, listening on the *same* port (or on different ports).

Thus, for example, to run one listener in a Production account and another in a Test account, on the same IP address/CPU, you must configure them to listen on different ports (e.g., 9200 for production and 9201 for Test). If, on the other hand, you are running the listeners on different IP addresses/CPUs, the ports can be the same (e.g., one Broker Listener on every system, listening on port 9200).

You need to configure your clients to connect to the appropriate listener port on your M server. While 9200 has been used as a convention for a Broker-based application service port, you can choose any available port greater than 1024 (sockets 1 to 1024 are reserved for standard, well-known services like SMTP, FTP, Telnet, etc.)

### Obtaining an Available Listener Port (for Alpha/VMS systems only)

Port selections conflict only if another process on the same system is using the same port. To list the ports currently in use on OpenVMS systems, use the DCL command:

```
$  UCX SHOW DEVICE_SOCKET
                                   Port                    Remote
Device_socket  Type    Local  Remote  Service           Host

  bg3         STREAM    5001       0  HL7              0.0.0.0
  bg23        STREAM    9700       0  Z3ZTEST          0.0.0.0
  bg24        STREAM    9600       0  ZSDPROTO         0.0.0.0
```

For example, if 9200 shows up in the Local Port column, some other application is already using this port number and you should choose another port.

## Starting And Stopping Listeners

### To Start All Listeners

To start all listeners (i.e., those configured in the RPC BROKER SITE PARAMETERS file [#8994.1] to automatically start), use the Start All RPC Broker Listeners option (introduced with Patch XWB*1.1*9). This option first **stops** any of these listeners that may be running, and then starts all of them up. TaskMan must be running.

DSM sites must run TaskMan in a DCL context, to start Listeners on any other node than the current ones. For more information on running TaskMan in a DCL context on DSM for OpenVMS, please refer to the "Kernel V. 8.0 Systems Manual."

### To Configure Listeners for Automatic Startup

To configure a given listener for startup by the Start All RPC Broker Listeners option, enter YES in the CONTROLLED BY LISTENER STARTER field in the RPC BROKER SITE PARAMETERS file (#8994.1) for that listener.

For more information, please refer to the "RPC BROKER SITE PARAMETERS File" topic in this chapter.

### To Stop All Running Listeners

To stop all running listeners (but, only those configured in the RPC Broker's site parameters to automatically start), use the Stop All RPC Broker Listeners option.

It is important to stop all Listeners before shutting down the system!

### To Start Up a Single Listener Directly

Enter the following at your M server's M prompt:

```
>D STRT^XWBTCP(Listener port)
```

### To Stop a Single Listener Directly

Enter the following at your M server's M prompt:

```
>D STOP^XWBTCP(Listener port)
```

If you want to restart this listener after stopping it, and other listeners are running on your system, start the listener up directly (see above) rather than via the Start All RPC Broker Listeners option (which first stops all listeners before restarting them).

### To Task the XWB LISTENER STARTER Option For System Startup

The XWB LISTENER STARTER option (which starts all configured Broker Listeners at one time). can be tasked to automatically start all of the Listener processes you need when TaskMan starts up, such as after the system is rebooted or configuration is restarted.

DSM sites must have TaskMan started via DCL, in order to use the XWB LISTENER STARTER option to automatically start Listener processes.

To automatically start the Listener(s) when TaskMan is restarted (i.e., in addition to the entries in the RPC BROKER SITE PARAMETERS file [#8994.1]), enter the XWB LISTENER STARTER option in the OPTION SCHEDULING file (#19.2). Schedule this option with SPECIAL QUEUING set to STARTUP. You can do this by using the TaskMan option: Schedule/Unschedule Options:

```
Select Systems Manager Menu Option: TASKMAN <Enter> Management

Select Taskman Management Option: SCH <Enter> edule/Unschedule Options

Select OPTION to schedule or reschedule: XWB LISTENER STARTER <Enter>  Start All
RPC Broker Listeners
         ...OK? Yes// <Enter>  (Yes)
      (R)

                        Edit Option Schedule
    Option Name:   XWB LISTENER STARTER
    Menu Text:    Start All RPC Broker Listeners      TASK ID:
________________________________________________________________________________

  QUEUED TO RUN AT WHAT TIME:

DEVICE FOR QUEUED JOB OUTPUT:

 QUEUED TO RUN ON VOLUME SET:

      RESCHEDULING FREQUENCY:

             TASK PARAMETERS:

            SPECIAL QUEUEING:  STARTUP

________________________________________________________________________________
```

**Figure 2-8: Automatically starting the Listener(s) when TaskMan is restarted**

## RPC BROKER SITE PARAMETERS File

The RPC BROKER SITE PARAMETERS file (#8994.1) contains one top-level entry, whose .01 field is a pointer to the DOMAIN file (#4.2). When the RPC Broker is installed, you create this top-level entry and assign the proper Domain Name.

The site parameters in this top-level entry pertain to listeners. For each listener that you plan to run on your system, you should make an entry for that listener in the site parameters.

### Editing the Listener Site Parameters

To create or edit listener entries, use the RPC Listener Edit option.

The RPC Listener Edit option first prompts you to select a Box-Volume Pair entry. Then, within each Box-Volume Pair entry (representing the volume set and system on which the listener should run), you can configure one or more listeners:

```
Select RPC BROKER SITE PARAMETERS DOMAIN NAME: YOURSITE.VA.GOV
         ...OK? Yes// <Enter>  (Yes)

Select BOX-VOLUME PAIR: KDE:ISC6A2// <Enter>
  BOX-VOLUME PAIR: KDE:ISC6A2// <Enter>
  Select PORT: 9500// <Enter>
    PORT: 9500// <Enter>
    STATUS: STARTING// <Enter>
    CONTROLLED BY LISTENER STARTER: YES//
```

**Figure 2-9: RPC Listener Edit option—Sample user dialogue**

The meaning of the site parameter fields for a given listener entry is as follows:

| Field | Meaning |
|---|---|
| BOX-VOLUME PAIR | Choose the Box-Volume pair representing one of the systems supporting "this" account, and on which a listener should run. |
| PORT | The port upon which the listener will listen. |
| STATUS | Ordinarily should not be edited (Use the Start All RPC Broker Listeners and Stop All RPC Broker Listeners options to start and stop listeners.) |
| CONTROLLED BY LISTENER STARTUP | If the listener should be started by the Start All RPC Broker Listeners option [XWB LISTENER STARTER], set this field to YES. Otherwise, set to NO. |

**Table 2-4: Listener site parameter entries description table**

## Integrated Auto Signon for Multiple User Sessions

Version 1.1 of the RPC Broker supports Kernel's Auto Signon from a client workstation to the server. Users need only sign on once (i.e., enter their Access and Verify codes) when accessing both a VistA roll-and-scroll (e.g., Lab, Pharmacy) and a VistA client/server GUI-based application (e.g., CPRS, NOIS, PCMM) on the same workstation, regardless of which application is started first. Once logged into the server, the user will *not* be asked to re-enter their Access and Verify codes for any subsequent VistA applications they may start.

Auto Signon is facilitated on the client side by the Broker Client Agent application (CLAGENT.EXE) and is only available for Telnet-based sessions in the roll-and-scroll environment.

### Enabling/Disabling Auto Signon

Control of the Auto Signon functionality is maintained and administered on the server for both VistA client/server applications (i.e., GUI) and the roll-and-scroll environment (i.e., terminal sessions). In support of that functionality, the DEFAULT AUTO SIGN-ON field was added to the KERNEL SYSTEM PARAMETERS file (#8989.3) and the AUTO SIGN-ON field was added to the NEW PERSON file (#200). The valid values for these fields are YES, NO, or DISABLED.

These fields, in conjunction with the other multiple signon fields, give the sites control of the implementation of Auto Signon for users in both the GUI and roll-and-scroll environments. The values in the AUTO SIGN-ON and MULTIPLE SIGN-ON fields in the NEW PERSON file (#200) take precedence over the values in the DEFAULT AUTO SIGN-ON and DEFAULT MULTIPLE SIGN-ON fields in the KERNEL SYSTEM PARAMETERS file (#8989.3). Therefore, the fields in the NEW PERSON file (#200) are checked first. If the user fields in the NEW PERSON file (#200) are null, the values in the KERNEL SYSTEM PARAMETERS file (#8989.3) will be used.

The AUTO SIGN-ON field in the NEW PERSON file (#200) and the DEFAULT AUTO SIGN-ON field in the KERNEL SYSTEM PARAMETERS file (#8989.3) are initially set to null.

**If a user is *not* allowed multiple signons, they will only be allowed to initiate a *single* session (i.e., automatically disallowing Auto Signon).**

#### Example 1:

If a user has an active VistA session and has the following characteristics:

- Allowed multiple signons (i.e., the MULTIPLE SIGN-ON field in the NEW PERSON file (#200) is set to YES)
- Allowed Auto Signon (i.e., the AUTO SIGN-ON in the NEW PERSON file (#200) is set to YES)

They will be allowed to start another VistA session *without* having to re-enter their Access and Verify codes.

**Example 2:**

If a user has an active VistA session and has the following characteristics:

- Allowed multiple signons (i.e., the MULTIPLE SIGN-ON field in the NEW PERSON file (#200) is set to YES)
- Not allowed Auto Signon (i.e., the AUTO SIGN-ON field in the NEW PERSON file (#200) is set to NO)

They will be allowed to start another VistA session, however, they *must* re-enter their Access and Verify codes.

The following table can be used as a guide to control multiple signons and Auto Signon for some typical situations:

| **Description** | [*] **User Settings** | [**] **System Settings** |
|---|---|---|
| **Multiple Signon:** | | |
| Disallow *all* users from having multiple signons | NO/NULL | NO |
| Allow *individual* users to have multiple signons | YES | NO |
| Allow *all* users to have multiple signons | YES/NULL | YES |
| **Auto Signon:**<br>ℹ With the exception for disabling Auto Signon, the following settings are only affective when users are allowed multiple signons. | | |
| Stop Auto Signon | Any Value | [†] DISABLED |
| Allow *individual* users to have Auto Signon | YES | NO |
| Disallow *individual* users from having Auto Signon | NO | YES |
| Allow *all* users to have Auto Signon | YES/NULL | YES |

**Table 2-5: Multiple and Auto Signon Settings table**

[*] User Settings refers to the NEW PERSON file (#200) and the following fields:

- MULTIPLE SIGN-ON (#200,200.04)
- AUTO SIGN-ON (#200,200.18)

**The User Settings override the **System Settings *except* when *disabling* Auto Signon!**

**System Settings refers to the KERNEL SYSTEM PARAMETERS file (#8989.3) and the following fields:

- DEFAULT MULTIPLE SIGN-ON (#8989.3,204)
- DEFAULT AUTO SIGN-ON (#8989.3,218)

†Sites may choose to disable Auto Signon (stops calls to the Broker Client Agent) for all users in the following situations:

- Network problems.
- Broker not installed.
- During installation of the Broker.

## RPC Broker Message Structure

The messages that are sent from a server to a client contain either several values or a single value. Presently, the RPC Broker messages are bound by the Microsoft Windows WINSOCK.DLL specifications and the size of the symbol table. The server receives a message from the client and parses out the name of the remote procedure call and its parameters. The Broker module on the server looks up the remote procedure call in the REMOTE PROCEDURE file (#8994) and executes the RPC using the passed parameters. At this point the server side of the application processes the request and returns the result of the operation. If the operation is a query, then the result is a set of records that satisfy that query. If the operation is to simply file the data on the server or it is unnecessary to return any information, then, typically, notification of the success of the operation will be returned to the client.

The basic RPC Broker message structure consists of the following:

- A header portion (which includes the name of the remote procedure call).
- The body of the message (which includes descriptors, length computations, and M parameter data).

## Client/Server Timeouts

The issue of timeouts is complex in a client/server environment. Because the user may be working with applications that rely solely on the client, long periods of time may elapse that the server would traditionally have counted against the user's timeout.

Broker Patch XWB*1.1*6 was created to address timeout issues. It instituted a "keep-alive" timer that was compiled into client applications. Through monitoring this keep-alive timer, the software is able to eliminate "ghost" server Broker jobs for which there is no longer a client application, based on the keep-alive timer rather than on user activity.

"Ghost" server jobs occur when client processes are ended in a non-standard way—for example, by pressing the PC's reset button. Prior to this patch, these jobs would wait for 10 hours to receive data from the client application that no longer existed.

In order to let the server know that the client application is still active, applications compiled with the client portion of Patch XWB*1.1*6 (and beyond) initiate a periodic, background contact with the server. This "polling" of the server by the client resets the timeout so that the server job is not stopped when the client still exists. Any client application compiled with the TRPCBroker and/or TSharedRPCBroker components distributed with the latest patch automatically polls. No developer or user intervention is necessary, and this polling activity affects neither the application nor the user.

The BROKER ACTIVITY TIMEOUT field in the KERNEL SYSTEM PARAMETERS file (#8989.3) controls the length of the timeout. That field was distributed by Kernel Patch XU*8.0*115 with a default value of approximately 3 minutes. By setting the timeout to a duration much shorter than 10 hours, the ghost jobs are eliminated quickly, if the client application is no longer running.

For advice regarding changing the value for this field, please refer to the help for the BROKER ACTIVITY TIMEOUT field.

The server portion of this patch is backwards compatible with client applications compiled with previous versions of the Broker. Thus, client applications do not have to be recompiled when this patch is installed on the server. The server retains a 10-hour timeout for those client applications compiled with previous Broker versions; that is, they continue to work as they did before the patch is installed.

The server side of this patch is effective only for client applications (like CPRS-GUI) that have been recompiled with the Broker Development Kit (BDK) portion of Patch XWB*1.1*6. Thus, installing the server patch alone does not eliminate the ghost jobs for client applications that have not been upgraded.

## Load Balancing on Alpha Systems

The Broker, like any Telnet or IP process, can be load balanced on DSM Alpha systems, if UCX 4.1 is running. The actual steps on configuring UCX for load balancing can be acquired from the ALPHA/AXP technical support group and will not be discussed here.

Multiple Broker servers can run on the same port as long as the machine IP addresses are unique. This is *not* a Broker requirement; it is a TCP/IP requirement. This capability is necessary for UCX load balancing. The multiple servers will receive a common alias that will be the connection destination.

In UCX, you should use the BIND alias:

For example:

```
UCX> show host vista.sitename.med.va.gov

     BIND database
```

```
Server:   999.999.999.xxx    999TNG

Host address      Host name

999.999.999.yy1   VISTA.SITENAME.MED.VA.GOV
999.999.999.yy2   VISTA.SITENAME.MED.VA.GOV
999.999.999.yy3   VISTA.SITENAME.MED.VA.GOV
999.999.999.yy4   VISTA.SITENAME.MED.VA.GOV
999.999.999.yy5   VISTA.SITENAME.MED.VA.GOV
```

**Figure 2-10: BIND alias in UCX**

In order to use load balancing, your client workstation needs to have DNS enabled and pointing to the IP address of the DNS server in the list. The Broker on the client will use the PC's DNS or HOSTS file to resolve the BROKERSERVER host name. In the previous example, the first DNS server is 999.999.999.xxx.

For example, if you want CPRS GUI to be "balanced," use the Edit Broker Servers program to edit the servers in the Microsoft Windows Registry and add in the alias VISTA.*sitename*.MED.VA.GOV.

For more information on adding servers to the Registry, please refer to the ""Connect To" Dialogue" or "Edit Broker Servers Program" topics in this chapter.

You don't want the alias in the HOSTS file because the HOSTS file is for static bindings only. If you want to put the alias in the HOSTS file, then you will have to make sure that the DNS server is first in the DNS list. Thus, when the user selects the BIND alias, the DNS will resolve it to one of the unique IP addresses and *not* to the HOSTS static assignment.

DSM sites do not need to have TaskMan started in a DCL context in order to use load balancing.

# 3. Security

## Security Features

Security in distributed computing environments, such as in client/server systems, is much more complicated than in traditional configurations. Although it is probably impossible to protect any computer system against the most determined and sophisticated intruder, the RPC Broker implements robust security that is transparent to the end user and without additional impact on IRM.

Security with the RPC Broker is a four-part process:

1. Client workstations must have a valid connection request.
2. Users must have valid Access and Verify codes.
3. Users must be valid users of a VistA client/server application.
4. Any remote procedure call must be registered and valid for the application being executed.

## Validation of Connection Request

An enhancement to security has been included with this version of the Broker. Before the Broker Listener jobs off a Handler for a client, it checks the format of the incoming connection request. If the incoming message does not conform to the Broker standard, the connection is closed. This serves as an early detection of impostors and intruders.

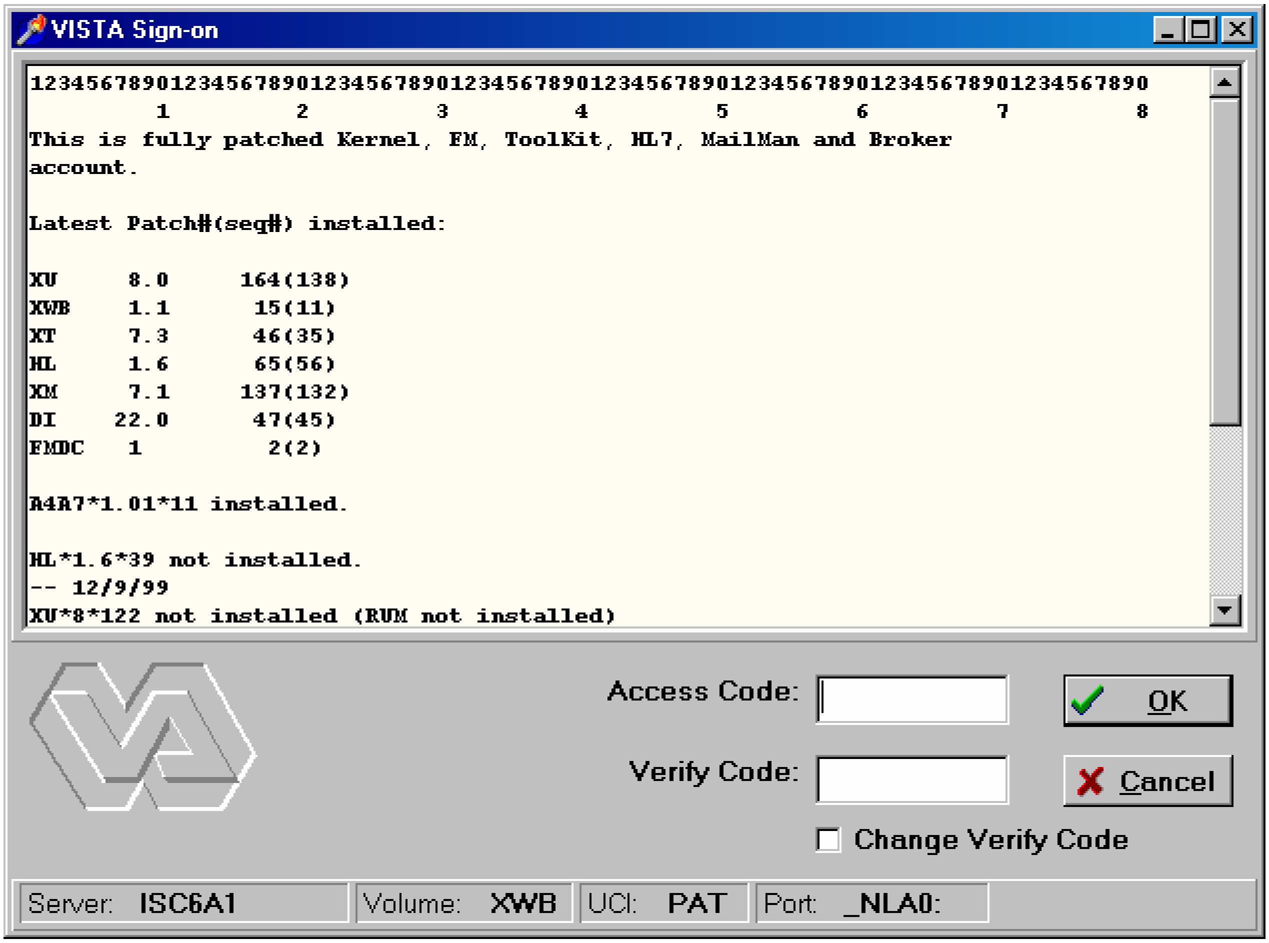
## Validation of Users

The GUI VistA Sign-on dialogue is integrated with the RPC Broker interface. This VistA Sign-on dialogue is invoked when the client application connects to the server.

### VistA Signon Dialogue

The VistA Signon dialogue automatically prompts users for their Access and Verify codes if they are not already signed on to a VistA application (see Figure 3-1).

A sample of the VistA Signon dialogue integrated with the RPC Broker is illustrated below:



```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
         1         2         3         4         5         6         7         8
This is fully patched Kernel, FM, ToolKit, HL7, MailMan and Broker
account.

Latest Patch#(seq#) installed:

XU      8.0      164(138)
XWB     1.1       15(11)
XT      7.3       46(35)
HL      1.6       65(56)
XM      7.1      137(132)
DI     22.0       47(45)
FMDC    1          2(2)

A4A7*1.01*11 installed.

HL*1.6*39 not installed.
-- 12/9/99
XU*8*122 not installed (RUM not installed)
```

Server: ISC6A1 | Volume: XWB | UCI: PAT | Port: _NLA0:

**Figure 3-1: Sample VistA Signon security dialogue**

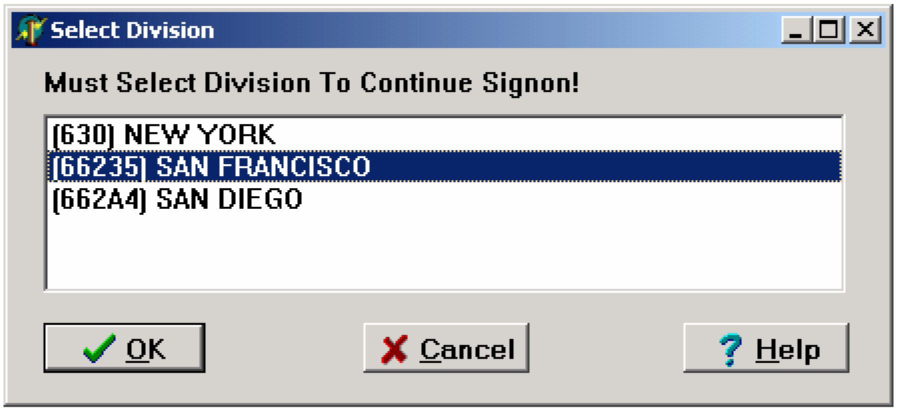
This version of the Broker supports both Auto Signon and Single Sign-On/User Context (SSO/UC).

For more information regarding Auto Signon, please refer to the "Integrated Auto Signon for Multiple User Sessions" topic in Chapter 2 in this manual or the description in Kernel Patch XU*8*59.

For more information on SSO/UC, please consult the *Single Sign-On/User Context (SSO/UC) Installation Guide* and *Single Sign-On/User Context (SSO/UC) Deployment Guide* on the VistA Documentation Library (VDL).

## VistA Division Selection Dialogue

After entering an Access and Verify code, if a user is associated with more than one Institution, the user will be presented with the following:



Must Select Division To Continue Signon!

[630] NEW YORK
[66235] SAN FRANCISCO
[662A4] SAN DIEGO

**Figure 3-2: Sample Select Division dialogue**

To continue the signon process, the user must select a division from the list presented. The user's default division will initially be highlighted. To choose a different division, users should click on or use the arrow keys to highlight the appropriate division and press the OK button after making their selection. The signon process will log the user into VistA with their DUZ(2) set to that division.

Client/server applications are "B"-type options (i.e., Broker options) in the OPTION file (#19). Users must have the client/server application option assigned to them like any other assigned option in VistA. It can be put on their primary menu tree or as a secondary option/menu as part of their suite of permitted options. The client/server application will only run for those users who are allowed to activate it.

The client/server application options will not be displayed in a user's menu tree.

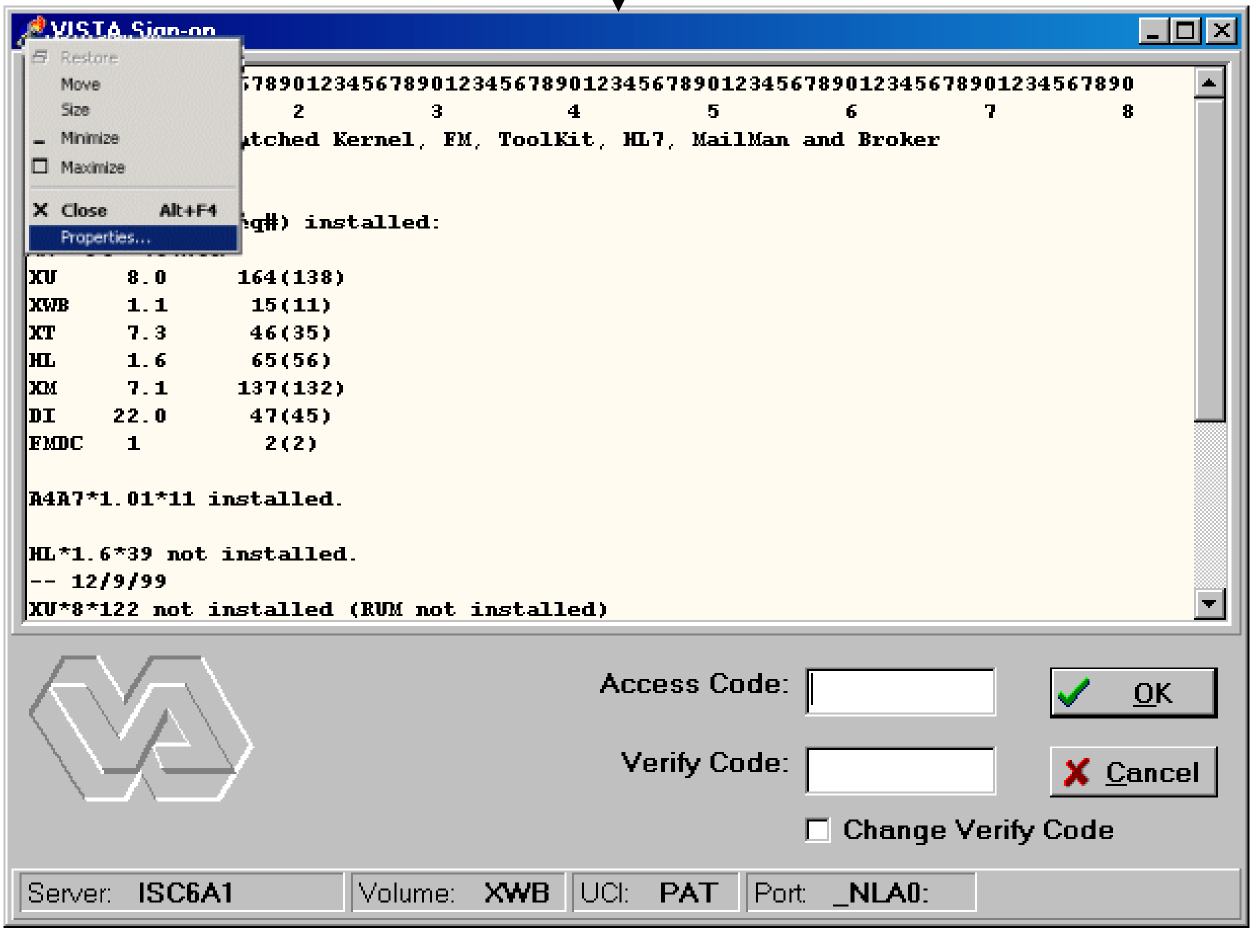
Kernel's Menu Manager verifies that users are allowed access to a VistA application or option with the following process:

1. Users start a VistA program.
2. The RPC Broker in the client application invokes the VistA Signon dialogue (see Figure 3-1) when connecting to the server.
3. Users sign on to the server via the Kernel signon process.
4. If authorized, the user is granted access to the server, otherwise an error message is returned. This serves as an initial security check.

For more information on Access and Verify codes or the Kernel signon process in general, please refer to the "Signon/Security" section in the *Kernel Systems Manual*.

## Users Can Customize VistA Signon Dialogue

When a VistA application on the client connects to the server, the VistA Sign-on dialogue is displayed for the user to identify and authenticate himself on the server. The VistA Sign-on dialogue System menu has a "Properties..." item, as shown below:



Move your mouse anywhere in the dialogue window's Title bar and right click (left click if you are left handed) to display the System menu.

```
5789012345678901234567890123456789012345678901234567890123456789 0
          2          3          4          5          6          7          8
atched Kernel, FM, ToolKit, HL7, MailMan and Broker


eq#) installed:

XU      8.0      164(138)
XWB     1.1       15(11)
XT      7.3       46(35)
HL      1.6       65(56)
XM      7.1      137(132)
DI     22.0       47(45)
FMDC    1          2(2)

A4A7*1.01*11 installed.

HL*1.6*39 not installed.
-- 12/9/99
XU*8*122 not installed (RUM not installed)
```

Access Code:

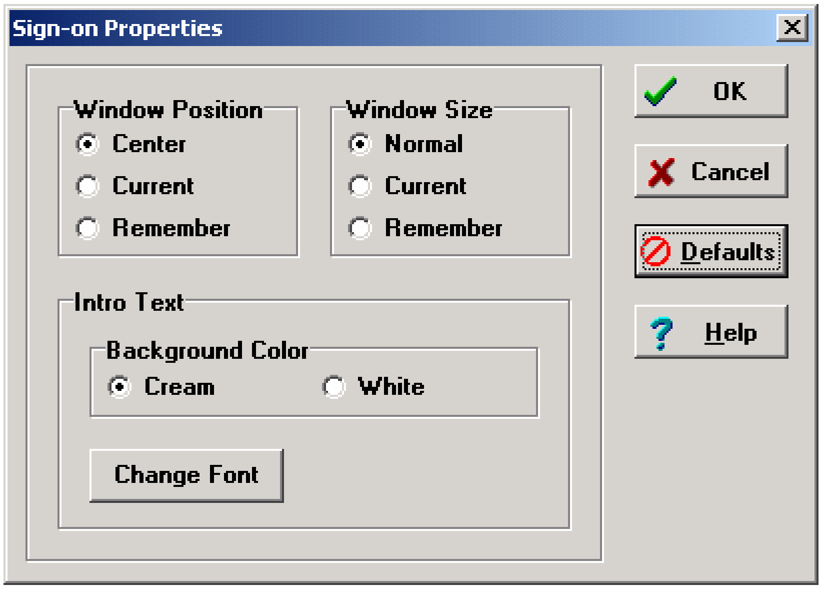
Verify Code:

☐ Change Verify Code

Server: ISC6A1 | Volume: XWB | UCI: PAT | Port: _NLA0:

**Figure 3-3: Sign-on Properties on the System Menu**

When this item is selected, the user is presented with the Sign-on Properties dialogue, as shown below:



Sign-on Properties

Window Position
● Center
○ Current
○ Remember

Window Size
● Normal
○ Current
○ Remember

Intro Text
Background Color
● Cream ○ White

**Figure 3-4: Sign-on Properties dialogue**

Using this form (see Figure 3-4), users can control the appearance of the VistA Sign-on dialogue by modifying the following characteristics:

- Window Position—The position of the VistA Sign-on dialogue.
- Window Size—The size of the VistA Sign-on dialogue.
- Introductory Text—The appearance of the introductory text in the VistA Sign-on dialogue.

### Window Position

The VistA Sign-on dialogue's window position can be one of the following:

| Position | Description |
|---|---|
| **Center** (default) | The VistA Sign-on dialogue will always appear in the center of the screen. |
| **Current** | The current position of the VistA Sign-on dialogue will be saved and used in the future. |
| **Remember** | Each time the VistA Sign-on dialogue is used and closed, it will record its position and open in that same place the next time it is used. |

**Table 3-1: Window position**

### Window Size

The VistA Sign-on dialogue's window size can be one of the following:

| Size | Description |
|---|---|
| **Normal** (default) | The size of the VistA Sign-on dialogue as it was designed. Typically, this is 500 pixels wide by 300 pixels high. |
| **Current** | The current size of the VistA Sign-on dialogue will be saved and used in the future. |
| **Remember** | Each time the VistA Sign-on dialogue is used and closed, it will record its size and open with the same size the next time it is used. |

**Table 3-2: Window size**

## Introductory Text

:

The VistA Sign-on dialogue's introductory text has a couple of settings users can control:

- **Background Color**:

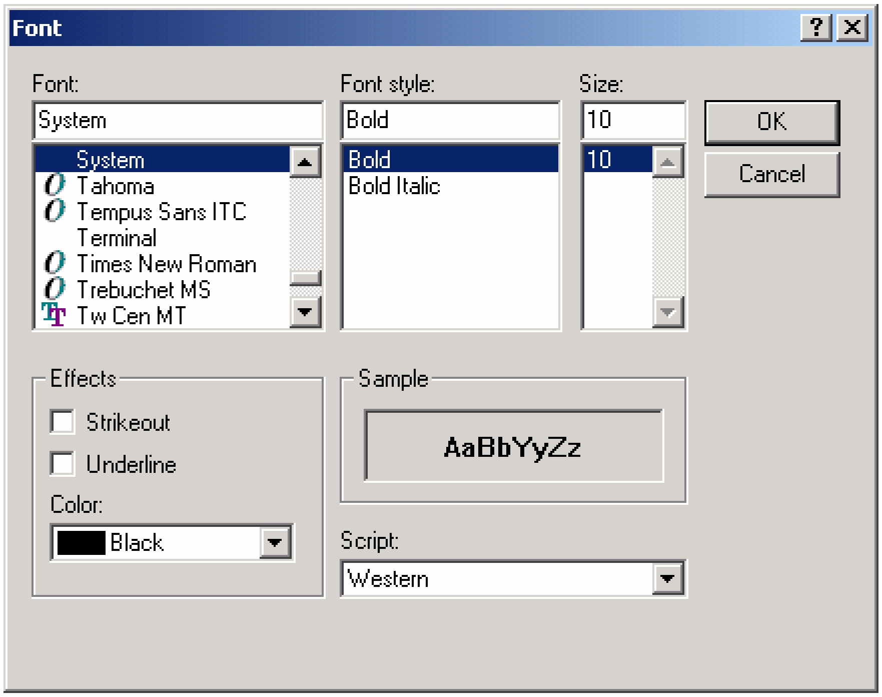
  :

| Color | Description |
|---|---|
| **Cream** (default) | According to the VA GUI conventions, this is the background color that should be used with text that users *cannot* edit. |
| **White** | For clarity and brightness. |

**Table 3-3: Introductory text background color**

- **Font**:

  When users press the "Change Font" button they are presented with a Font form that can be used to change the font face, style, size, effects, and color of the introductory text of the VistA Sign-on dialogue, as shown below:



Font

Font: System

Font style: Bold

Size: 10

Effects

☐ Strikeout

☐ Underline

Color: Black

Sample

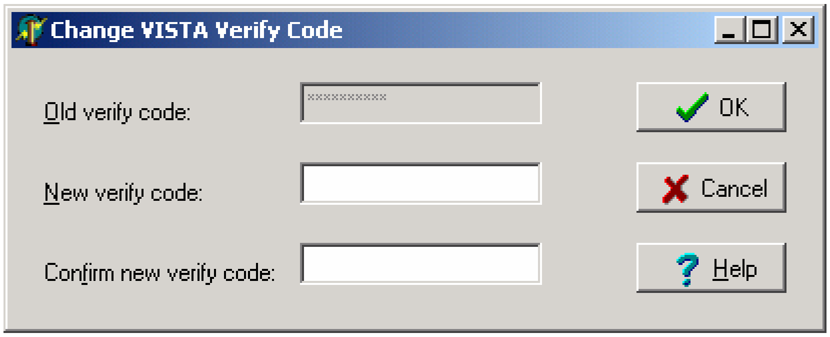
AaBbYyZz

Script: Western

**Figure 3-5: Sample Font dialogue**

## Change VistA Verify Code Component

Version 1.1 of the Broker includes a Change VistA Verify Code dialogue for the client workstation. After a user signs onto the server, if their Verify code has expired, the user is automatically prompted with the following message: "You must change your Verify code at this time." Once the user presses the OK button they are presented with the Change VistA Verify Code dialogue as displayed below:



Change VISTA Verify Code

Old verify code: ***********

New verify code:

Confirm new verify code:

**Figure 3-6: Change VistA Verify Code dialogue**

The old Verify code will appears as asterisks (*****) in a grayed-out box.

Users must then do the following:

- Enter their new Verify code.
- Confirm their new Verify code.

Users who wish to change their Verify code prior to its expiration can do so by either of the following methods:

- GUI environment (available as of Broker Patch XWB*1.1*13)—Click on the checkbox labeled "Change VC" on the Sign-on screen (see Figure 3-1). After signing on, it invokes the dialogue described above (see Figure 3-6).
- Roll-and-Scroll environment (existing functionality)—Use the Edit User Characteristics option [XUSEREDITSELF] to edit your Verify code.

## Validation of RPCs

The RPC Broker security allows any RPC to run when it is properly registered to the VistA client/server application. The Broker on the server along with Kernel's Menu Manager determines which application a user is currently running. Menu Manager determines if a user is allowed to run this application or option by the following process:

1. A remote procedure call is sent by a client application and is received by the RPC Broker on the server.
2. The Broker verifies that the RPC is "registered" to the application that the user is currently running, *prior* to executing the remote procedure call (RPC).

   The application being run is designated by a "B"-type option in the OPTION file (#19). The application must specify the option and that option *must* be in a user's menu tree.

   For more information on registering an RPC to a VistA software application, please refer to the "RPC Security: How to Register An RPC" topic in the *RPC Broker Getting Started with the Broker Development Kit (BDK)* manual.
3. Menu Manager checks if the RPC is registered for this software option. If not properly registered, Menu Manager will return a message explaining why the RPC is not allowed.
4. The Broker on the server executes the RPC if it is registered, otherwise it is rejected.

## Sample Security Procedures

The security steps each client user will follow and the intermediate client/server security processes are described in the following example:

| Step | Description |
|---|---|
| **1.** | The user starts a VistA program on the client. For this example, the user clicks on the Computerized Patient Record System (CPRS) application icon. |
| **2.** | The user must sign on to the server through the VistA Sign-on dialogue (see Figure 3-1) on the client using their Access and Verify codes invoking the Kernel signon process. |
| **3.** | The Menu Manager on the server verifies the user is allowed access to the "B"-type option requested by CPRS. |
| **4.** | The Menu Manager on the server verifies the option is a "client/server" type option and the requested RPC is in that option's RPC multiple. |
| **5.** | If all of the previous steps complete successfully, the application RPC is launched. |

**Table 3-4: Sample security procedures**

## Security Features Tasks Summary

The following table summarizes required security tasks:

| Security Task | Completed By |
|---|---|
| Verify valid connection request | RPC Broker |
| Verify valid user | Kernel Signon |
| Verify user is authorized to run this software | RPC Broker & Menu Manager |
| Verify an RPC is registered to an application | RPC Broker & Menu Manager |
| Application—RPC Registration | KIDS |

**Table 3-5: Security tasks summary**

To reiterate, an RPC is only allowed to run within the context of an application with which it is registered. Users are only able to run the server side of the application that was installed on the server by IRM.

**For each release of the RPC Broker, the RPC Broker Development Team will continuously strive to implement the most complete, robust, and flexible security available at the time.**

# 4. Troubleshooting

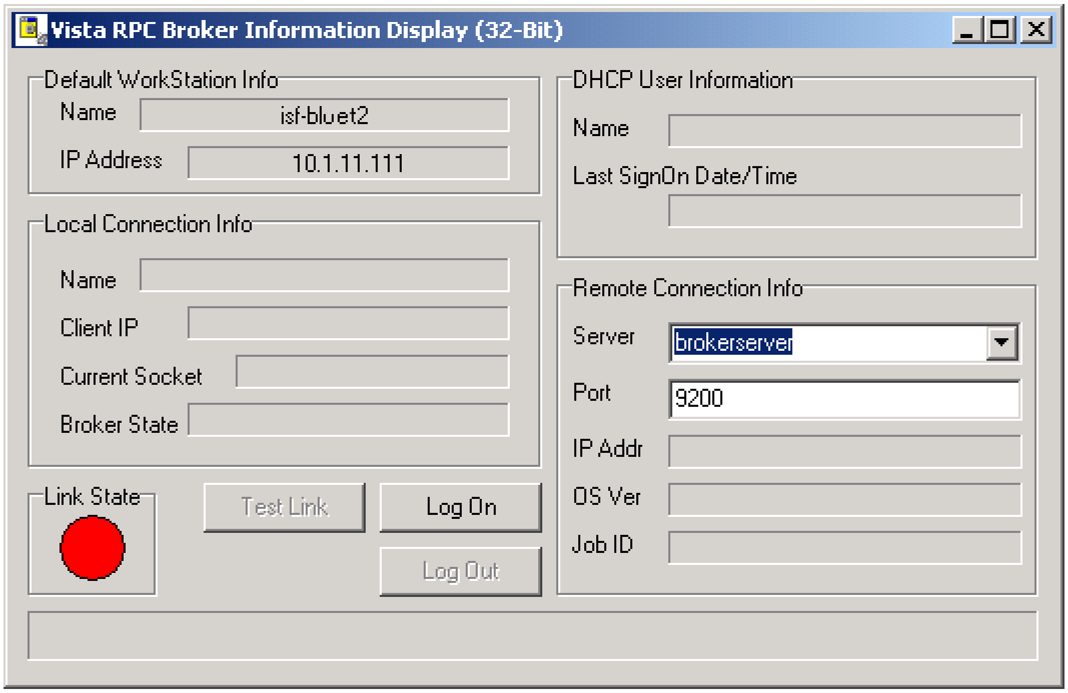
## Test the Broker Using the RPC Broker Diagnostic Program

This version of the Broker includes a diagnostic tool for the client workstation (see Figure 4-1). This tool can be used to verify and test the Broker client/server connection and signon process. This program (i.e., RPCTEST.EXE) also displays specific information about the client workstation that can be useful to IRM personnel when trying to determine and/or correct any problems with or to test the Broker.

It displays the following information:

- Default workstation information that includes the Name and IP Address.
- Local connection information that includes the Name, Client IP, Current Socket, and Broker State.
- VistA user information that includes the Name and Last Signon Date/Time.
- Remote connection information that includes the Server, Port, IP Address, Operating System Version information, and Job ID.
- A color-coded Link State indicator that shows the status of your connection:
  - ➢ Red = no link/connection.
  - ➢ Yellow = attempting link/connection.
  - ➢ Green = successful link/connection.

When you run the RPC Broker Connection Diagnostic Program (i.e., RPCTEST.EXE), the following dialogue will be displayed:



Name isf-bluet2

IP Address 10.1.11.111

Server brokerserver

Port 9200

**Figure 4-1: RPC Broker connection diagnostic program**

You should verify that the connection from the client workstation to the server is functioning correctly. For example:

- Try logging on to the server by choosing a server/port combination and pressing the "Log On" button; you will be presented with the VistA Sign-on dialogue. The Link State indicator will change from red to yellow to green as you progress through the connection process.
- Test various connections by changing the server and port information under the "Remote Connection Info" block. To verify the connection process is working properly, try logging on to known servers and ports with Listeners running.

You can also use this tool to resolve a server address without having to log on to the server. Type in a server name in the "Server" box located in the Remote Connection Info section of the dialogue and press the enter key. If the server can be found, the IP address will be displayed in the "IP Addr" box in that same section.

If you encounter an error while testing the Broker, make sure you check the following:

- Is the Broker Listener running on the specified port? If not, start the Broker Listener on the specified port.

For more information on starting the Broker Listener, please refer to the "Broker Listeners and Ports" topic in Chapter 2 of this manual.

- Have you installed all current Kernel, Kernel Toolkit, and VA FileMan patches? If not, you must install all required patches (see the *RPC Broker Installation Guide*).
- Did you change the IP address for BROKERSERVER in the HOSTS file in this session? If the IP address and server name are not resolvable, you need to correct the entry.

> Your site can use the HOSTS file or DNS to resolve IP addresses and server names. If the HOSTS file is not supported in your LAN, then you will need to work with the DNS database and see if the value returned by the DNS query really identifies the machine where the listener is running.

- Is the IP address resolvable for the BROKERSERVER listed under the TCP/IP Server? If not, edit the HOSTS file in your Microsoft Windows directory and correct the IP address for the BROKERSERVER or resolve the IP address with DNS.
- Does the TCP/IP address (used in the HOSTS file) correspond to the IP address that is owned by the node used to start up the Broker Listener? If you have several nodes that can service your Test/Production account, you must make sure that the one used to start up the Listener is the one being referenced in the HOSTS file.

## Verify and Test the Network Connection

To detect and avoid network problems, do the following:

1. First, make sure you actually have TCP/IP running correctly on your workstation.

   At the DOS/Command prompt type PING nnn.nnn.nnn.nnn to the server host to which you are trying to connect (where nnn.nnn.nnn.nnn equals the IP address of the server). For example:

   ```
   C:\>PING 127.0.0.1
   ```

   Alternatively, you can PING the same server name you are trying to connect to or resolve (e.g., BROKERSERVER). For example:

   ```
   C:\>PING BROKERSERVER
   ```

   > "PINGing" is a way to test connectivity. PINGing sends an Internet Control Message Protocol (ICMP) packet to the server in question and requests a response. It verifies that the server is running and the network is properly configured.

   - If the host is unreachable, there is a network problem and you should consult with your network administrator.
   - If you get a timeout, it may be your network configuration on the client workstation, proceed to Step #2.
   - If the server is reachable, proceed to Step #4.

2. Check the properties of the WINSOCK.DLL on the client workstation and make sure it's the correct version. Install the latest Service Pack.
3. Make sure that the files on the client are in the correct directories.

In Microsoft Windows 95, the WINSOCK.DLL expects the HOSTS file to be located in the WINDOWS root directory. You should only have one copy each of the WINSOCK.DLL and the HOSTS file on the client. (However, there may be a second copy that WIN95 keeps in the WINDOWS/SYSBCKUP directory). If Windows 95 detects that some of its core files have been overwritten with older versions, supposedly it will automatically update files on reboot.

4. Make sure that all of the client workstation TCP/IP settings are correct in the network properties. Typo's, etc. can be a real problem, as can gateways, DNS servers, etc. Try removing items in your WINS configuration/DNS configuration, etc.

For more information on telecommunications support, please visit the Telecommunications Support Office Home Page at the following address:

http://vaww.va.gov/cso/

## Signon Delays

Users signing on to VistA on a client workstation with the Broker Client Agent running should *not* experience any signon delays.

In order to provide users with the capability of Auto Signon in both a GUI and roll-and-scroll Telnet session, the Kernel signon process was modified.

The Kernel signon process now tries to contact the RPC Broker V. 1.1 Client Agent on the client workstation (i.e., prior to and following the Access and Verify code prompts) to allow Auto Signon to take place. A three-second (or less) delay is built into this process while attempting to connect to the Client Agent and allow for any possible network delays.

If you wish to eliminate the 3-second (or less) signon delay in a GUI/Telnet session (i.e., *not* associated with network delays), do either of the following:

1. Disable Auto Signon for *all* users by setting the DEFAULT AUTO SIGN-ON field in the KERNEL SYSTEM PARAMETERS file (#8989.3) to "DISABLED"
2. Install and run the Broker Client Agent on *all* client workstations, if Auto Signon is enabled on your system.

For more information on the DEFAULT AUTO SIGN-ON field, please refer to the "Integrated Auto Signon for Multiple User Sessions" topic in Chapter 2 in this manual.

## RPC BROKER FAQS

For examples of general or development-specific frequently asked questions (FAQs) about the RPC Broker, please refer to the following Web site:

http://vista.med.va.gov/broker/faqs.asp

# Glossary

CLIENT

A single term used interchangeably to refer to the user, the workstation, and the portion of the program that runs on the workstation. In an object-oriented environment, a client is a member of a group that uses the services of an unrelated group. If the client is on a local area network (LAN), it can share resources with another computer (server).

COMPONENT

An object-oriented term used to describe the building blocks of GUI applications. A software object that contains data and code. A component may or may not be visible. These components interact with other components on a form to create the GUI user application interface.

DHCP

**D**ynamic **H**ost **C**onfiguration **P**rotocol.

DLL

**D**ynamic **L**ink **L**ibrary. A DLL allows executable routines to be stored separately as files with a DLL extension. These routines are only loaded when a program calls for them. DLLs provide several advantages:

1. DLLs help save on computer memory, since memory is only consumed when a DLL is loaded. They also save disk space. With static libraries, your application absorbs all the library code into your application so the size of your application is greater. Other applications using the same library will also carry this code around. With the DLL, you don’t carry the code itself, you have a pointer to the common library. All applications using it will then share one image.

2. DLLs ease maintenance tasks. Because the DLL is a separate file, any modifications made to the DLL will not affect the operation of the calling program or any other DLL.

3. DLLs help avoid redundant routines. They provide generic functions that can be utilized by a variety of programs.

GUI

**G**raphical **U**ser Interface. A type of display format that enables users to choose commands, initiate programs, and other options by selecting pictorial representations (icons) via a mouse or a keyboard.

ICON

A picture or symbol that graphically represents an object or a concept.

REMOTE PROCEDURE CALL

A remote procedure call (RPC) is essentially M code that may take optional parameters to do some work and then return either a single value or an array back to the client application.

SERVER

The computer where the data and the Business Rules reside. It makes resources available to client workstations on the network. In VistA, it is an entry in the OPTION file (#19). An automated mail protocol that is activated by sending a message to a server at another location with the "S.server" syntax. A server's activity is specified in the OPTION file (#19) and can be the running of a routine or the placement of data into a file.

USER ACCESS

This term is used to refer to a limited level of access to a computer system that is sufficient for using/operating software, but does not allow programming, modification to data dictionaries, or other operations that require programmer access. Any of VistA 's options can be locked with a security key (e.g., XUPROGMODE, which means that invoking that option requires programmer access).

The user's access level determines the degree of computer use and the types of computer programs available. The Systems Manager assigns the user an access level.

USER INTERFACE

The way the software is presented to the user, such as Graphical User Interfaces that display option prompts, help messages, and menu choices. A standard user interface can be achieved by using Borland's Delphi Graphical User Interface to display the various menu option choices, commands, etc.

WINDOW

An object on the screen (dialogue) that presents information such as a document or message.

For a comprehensive list of commonly used infrastructure- and security-related terms and definitions, please visit the ISS Glossary Web page at the following Web address:

http://vista.med.va.gov/iss/glossary.asp

For a comprehensive list of acronyms, please visit the ISS Acronyms Web site at the following Web address:

http://vista/med/va/gov/iss/acronyms/index.asp

# Index